ST-DenNetFus: A New Deep Learning Approach for Network Demand Prediction

Haytham Assem$^{1,2[0000-0002-9475-0579]}$, Bora Caglayan$^{1[0000-0002-8491-4453]}$, Teodora Sandra Buda$^{1[0000-0003-1806-2915]}$, and Declan O'Sullivan$^{2[0000-0003-1090-3548]}$

[1] Cognitive Computing Group, Innovation Exchange, IBM, Ireland
{haythama@ie., tbuda@ie., bora.caglayan@}ibm.com

[2] School of Computer Science & Statistics, Trinity College Dublin, Ireland
declan.osullivan@cs.tcd.ie

Abstract. Network Demand Prediction is of great importance to network planning and dynamically allocating network resources based on the predicted demand, this can be very challenging as it is affected by many complex factors, including spatial dependencies, temporal dependencies, and external factors (such as regions' functionality and crowd patterns as it will be shown in this paper). We propose a deep learning based approach called, ST-DenNetFus, to predict network demand (i.e. uplink and downlink throughput) in every region of a city. ST-DenNetFus is an end to end architecture for capturing unique properties from spatio-temporal data. ST-DenNetFus employs various branches of dense neural networks for capturing temporal closeness, period, and trend properties. For each of these properties, dense convolutional neural units are used for capturing the spatial properties of the network demand across various regions in a city. Furthermore, ST-DenNetFus introduces extra branches for fusing external data sources that have not been considered before in the network demand prediction problem of various dimensionalities. In our case, these external factors are the crowd mobility patterns, temporal functional regions, and the day of the week. We present an extensive experimental evaluation for the proposed approach using two types of network throughput (uplink and downlink) in New York City (NYC), where ST-DenNetFus outperforms four well-known baselines.

Keywords: Spatio-temporal Data, Deep Learning, Convolutional Neural Networks, Dense Networks, Network Demand Prediction

1 Introduction

Mobile data traffic has increased dramatically in the last few decades [8]. Besides, the increase in the number of devices accessing the cellular network, emerging social networking platforms such as Facebook and Twitter has further added to the mobile data traffic [10]. This led to the need to increase the network resources provided to end-users and consequently this has caused a huge cost

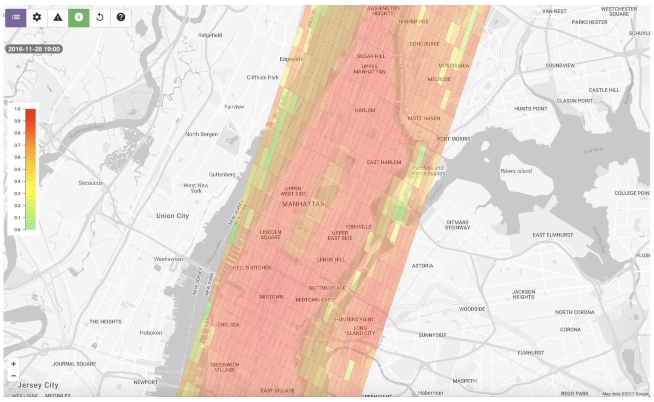


Fig. 1: Network Demand Prediction using the introduced ST-DenNetFus approach.

increase on the operators. The mobile network operators are striving for solutions to reduce the OPEX and CAPEX costs of such emerging demand. Reducing the OPEX and CAPEX cost is not only of importance to the operators but as well to the environment. The statistics show that the total CO2 emissions from the information and communication technology (ICT) infrastructure contributes for 2% of total CO2 emissions across the globe in which the telecommunication industry is a major part of it [15]. There are significant spatial and temporal variations in cellular traffic [24] and the cellular system is designed using the philosophy of worst case traffic (such as to fulfil the quality of service (QoS) in case of a peak traffic). Hence, there is a growing need to have a spatio-temporal prediction based model for the Network Demand Prediction problem [26].

In this paper, two types of cellular network throughput, downlink and uplink are predicted using a new deep learning based-approach called ST-DenNetFus (see Fig. 1): i) Downlink is the total downloaded network throughput in a region during a given time interval. ii) Uplink denotes the total uploaded network throughput in a region during a given time interval. Both types of throughput track the overall Network Demand[3] of a region.

Deep learning has been applied successfully in many applications, and is considered one of the most cutting edge techniques in Artificial Intelligence (AI) [18]. There are two types of deep neural networks that try to capture spatial and temporal properties: a) Convolutional Neural Networks (CNNs) for capturing spatial structure and dependencies. b) Recurrent Neural Networks (RNNs) for learning temporal dependencies. However, (and similar to the crowd flow prediction problem in the urban computing domain presented in [37]) it is still very challenging to apply these type of techniques to the spatio-temporal Network Demand Prediction problem due to the following reasons: **(A) Spatial dependencies: (A.1) Nearby -** The downlink throughputs of a region might be

[3] In this paper terminology, we refer to both types of throughput, uplink and downlink as “Network Demand”.

affected by the uplink throughputs of nearby regions and vice versa. In addition, the downlink throughputs of a region would affect its own uplink throughputs as well. **(A.2) Distant** - The network demand of a region can be affected by the network demand of distant regions especially if both are supported by same Base Station geographically. **(B) Temporal dependencies: (B.1) Closeness** - Intuitively, the network demand of a region is affected by recent time intervals. For instance, a high network demand occurring due to a crowded festival occurring at $9PM$ will affect that of $10PM$. **(B.2) Period** - Network demand during morning or evening hours may be similar during consecutive weekdays, repeating 24 hours. **(B.3) Trend** - Network demand may increase as summer approaches especially on weekends. Recent study [32] showed that the summer usage increases in the evening and early morning hours from about midnight to $4AM$, which indicates that young adults are not putting down mobile devices just because it is a summer break. **(C) External Factors: (C.1) Multiple** - Some external factors may impact the network demand such as the temporal functional regions, crowd mobility patterns and day of the week. For example, a business functional region may rely on the wireless networks more than the cellular networks. In addition, a highly crowded area has a higher chance for more cellular network usage. **(C.2) Dimensionality** - The external factors may vary in the dimensionality of the data. For example, the day of the week data will be in 1-dimensional space since it varies across time only but crowd mobility or temporal functional regions data will be in 2-dimensional space since it varies across space and time.

To tackle the above challenges, in this paper a spatio-temporal deep learning based-approach called ST-DenNetFus is proposed that collectively predict the uplink and downlink throughputs in every region in a city. The proposed contributions in this paper are five-fold:

1. ST-DenNetFus employs convolutional-based dense networks to model both nearby and distance spatial dependencies between regions in cities.
2. ST-DenNetFus employs several branches for fusing various external data sources of different dimensionality. The ST-DenNetFus architecture proposed is expandable according to the availability of the external data sources needed to be fused.
3. ST-DenNetFus uses three different dense networks to model various temporal properties consisting of temporal closeness, period, and trend.
4. The proposed approach has been evaluated on a real network data extracted from NYC and in particular Manhattan, for 6 months. The results reinforce the advantages of the new approach compared to 4 other baselines.
5. For the first time, it is shown that extracted urban patterns that have not been considered before in this particular problem (such as crowd mobility patterns and temporal functional regions) when fused as an external data sources for estimating the network demand, lead to more accurate prediction results.

2 Related Work

In this section, the state-of-the-art is reviewed from two different perspectives. First, an overview on the Network Demand Prediction related work is presented and then an overview on the recent advancements of convolutional neural networks is presented.

Cellular network throughput prediction plays an important role in network planning. To name a few works, in [9], ARIMA and exponential smoothing model are used for predicting the network demand for a single cell and whole region scenarios. ARIMA was found to outperform for a whole region scenario while the exponential smoothing model had better performance for the single cell scenario. In [35], a hybrid method using both ARMA and FARIMA is introduced to predict the cellular traffic where FARIMA found to work effectively on the time series that hold long range dependence. For long time prediction, the authors in [25] presented an approach with 12-hour granularity that allows to estimate aggregate demands up to 6 months in advance. Shorter and variable time scales are studied in [27] and [39] adopting ARIMA and GARCH techniques respectively. Recently, researchers started to exploit external sources trying to achieve more reliable and accurate predictions. In [1], the authors propose a dynamic network resources allocation framework to allocate downlink radio resources adaptively across multiple cells of 4G systems. Their introduced framework leverages three types of context information: user's location and mobility, application-related information, and radio maps. A video streaming simulated use case is used for evaluating the performance of the proposed framework. Another interesting work presented in [34] focuses on building geo-localized radio maps for a video streaming use-case in which the streaming rate is changed dynamically on the basis of the current bandwidth prediction from the bandwidth maps. *To the best our knowledge, in the field of telecommunications, for the first time end-to-end deep learning with fusing urban patterns as external data sources is considered, showing the effectiveness of the proposed deep learning approach along with the impact of some of the urban patterns in achieving higher accuracy for predicting network demand.*

In the last few years, deep learning has led to very good performance on a variety of problems, such as visual recognition, speech recognition and natural language processing [23] as well as spatio-temporal prediction problems [4] and environmental challenges [3]. Among different types of deep neural networks, convolutional neural networks (CNN) have been most extensively studied. Since 2006, various methods have been developed to overcome the limitations and challenges encountered in training deep CNNs. The most notable work started by Krizhevsky et al. when they introduced an architecture called AlexNet [17]. The overall architecture of AlexNet is similar to LeNet-5 but with deeper structure and showed significant improvements compared to LeNet-5 on the image classification task. With the success of AlexNet, several successful architectures have evolved, ZFNet [36], VGGNet [30], GoogleNet [31] and ResNet [11]. One of the main typical trends with these evolving architectures is that the networks are getting deeper. For instance, ResNet, the winner of ILSVRC 2015 competition got deeper

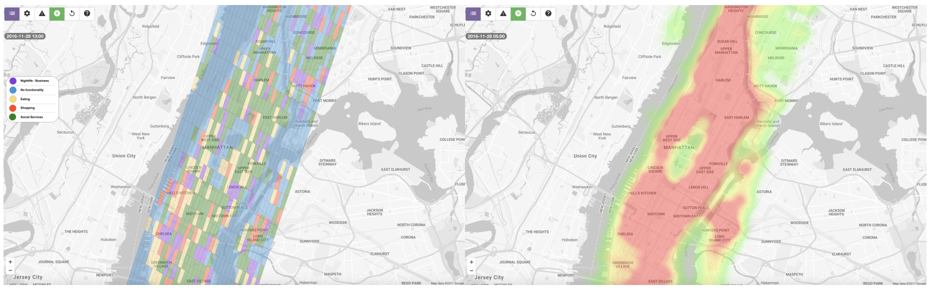

(a) The extracted temporal functional regions. (b) The extracted crowd mobility patterns

Fig. 2: External factors extraction.

20 times more deeper than AlexNet and 8 times deeper than VGGNet. This typical trend is because networks can better approximate the target function when they are deeper. However, the deeper the network the more complex it is, which makes it more difficult to optimize and easier to suffer overfitting. Recently in 2016, a new architecture has been introduced called DenseNets [13] that exploits the potential of the network through feature reuse, yielding condensed models that are easy to train and highly parameter efficient. DenseNets obtain significant improvements over most of the state-of-the-art networks to date, whilst requiring less memory and computation to achieve high performance [13]. *Hence, in this work we rely mainly on leveraging the dense blocks as a core part of the proposed ST-DenNetFus architecture as will be described in the sections to follow. To the best our knowledge, this is the first work to show the effectiveness of DenseNet on a different domain than computer vision.*

3 External factors extraction

In this section, our approach for extracting the considered three external factors that are fused in the ST-DenNetFus architecture are described.

For the first external factor called *temporal functional regions* and as defined in [2] [5], a temporal functional region is defined as “a set of regions that change their functionality over space and time, depending on the activity shifts across the temporal variation of the different time slots considered.” The approach introduced for extracting these temporal functional regions was based on using location-based social networks (LBSNs) data and applying clustering techniques based on the time-slot. However, in this paper we propose a different approach for extracting temporal functional regions based on Point-Of-Interests (POIs) data. We argue that our proposed approach is more advantageous as accessing POIs data is easier compared to LBSNs data. We extracted the POIs in Manhattan region using the Google Places API[4] after finding various inconsistencies in the openly accessible Open Street Maps data. We take into consideration the opening and closing times of the point of interest to increase the accuracy of the functionality mapping. If opening and closing time of a particular POI was not available, we used the mean opening and closing times for that category. The functionality of a point of interest can be: 1) Education, 2) Business, 3) Shopping, 4) Nightlife, 5) Eating, 6) Social services and this information is inferred from the POI description accessed through Google Places API. We count the active point of interest counts for each hour and for each function to generate this particular external factor. The output of this external factor is six 2D-dimensional matrices where each corresponds to a different functionality based on the POI frequency count with shape $(N * M * T)$. Fig. 2a shows an example of extracted functional regions during certain time-slot in a day across the defined 32x32 grid map.

For the second external factor related to *crowd mobility patterns*, we calculated the crowd count in our experiments based on the unique anonymized identifiers (*ueid*) of the users in the mobile network sessions available in the network dataset. The crowd count is the number of unique *ueid* in a particular region in an hour. A user that travels between regions in the period is assumed to be present in multiple regions in that hour. The output of this external 2D-dimenional external factor follows a shape of $(N * M * T)$. Fig. 2b shows an example of a crowd mobility pattern during certain time-slot during the day across the defined 32x32 grid map visualized using heat-map.

The third and final external factor that we took into account when predicting network demand is related to the day of the week which for each sample in the dataset, we inferred the corresponding day of the week as well as whether this day is weekend or not. This resulted in a 1D-dimensional external factor that change across time.

4 Deep Spatio-Temporal Dense Network with Data Fusion (ST-DenNetFus)

Fig 3 presents the proposed architecture of ST-DenNetFus where each of the downlink and uplink network throughputs at time t is converted to a 32×32 of 2-channel image-like matrix spanning over a region. Then the time axis is divided into three fragments denoting recent time, near history and distant history. Further, these 2-channel image-like matrices are fed into three branches on the right side of the diagram for capturing the trend, periodicity, and closeness and output $\mathbf{X}_{in}$. Each of these branches starts with convolution layer followed by L dense blocks and finally another convolution layer. These three convolutional based branches capture the spatial dependencies between nearby and distant regions. Moreover, there are number of branches that fuse external factors based on their dimensionality. In our case, the temporal functional regions and the crowd mobility patterns are 2-dimensional matrices $(\mathbf{X}_{Ext-2D})$ that change across

[4] https://developers.google.com/places/

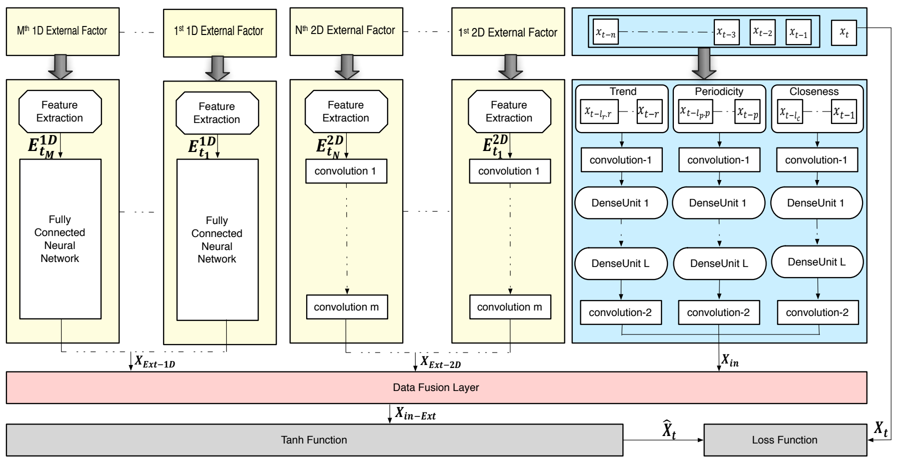


Fig. 3: ST-DenNetFus Architecture.

space and time but on the other side, the day of the week is 1-dimensional matrix that change across time only ($\mathbf{X}_{Ext-1D}$). At that stage a data fusion layer is introduced that fuses the $\mathbf{X}_{in}$, $\mathbf{X}_{Ext-2D}$, and $\mathbf{X}_{Ext-1D}$. The output is $\mathbf{X}_{in-Ext}$ which is fed to *tanh* function to be mapped to $[-1, 1]$ range. This helps towards a faster convergence in the backpropagation learning compared to a standard logistic function [19].

4.1 Network Throughput input data

The network throughput data for both uplink and downlink are fed into the first three branches (shown in blue in Fig 3).

Convolution Design. Since a city usually has a very large size with many regions, and intuitively the network demand may be affected by nearby as well as distant regions, convolutional neural network can handle this effectively as it captures the spatial structure through convolutions [38]. In order to capture the dependency between regions, there is a need to design many convolutional layers. Subsampling techniques have been introduced to preserve distant dependencies and avoid the loss of resolution especially in video sequence generating tasks [21]. Unlike with the common approach to CNN, we do not use subsampling but instead rely only on convolutions [14]. Support for such an approach can be found in [38], where the authors were trying to capture the spatial dependencies at a citywide scale similar to our problem here. They concluded that one convolution naturally captures spatial near dependencies, and a stack of convolutions afterwards can further capture the spatial distant citywide dependencies. The closeness, periodicity, and trend components adapt 2-channel image-like matrices according to the time interval as follows, $[\mathbf{X}_{t-l_c}, ..., \mathbf{X}_{t-1}]$, $[\mathbf{X}_{t-l_p \cdot p}, ..., \mathbf{X}_{t-p}]$, and

$[\mathbf{X}_{t-l_r.r}..., \mathbf{X}_{t-r}]$ respectively. l_c, l_p and l_r represent the length of the dependent sequence for the closeness, period and trend while c, p and r depicts their span respectively. In our detailed implementation, the p captures the daily periodicity while the r is equal to one week that reveals the weekly trend of the network demand.

Each of these inputs is concatenated across the first axis (time interval) as tensors, $\mathbf{X}_c^{(0)}$, $\mathbf{X}_p^{(0)}$, and $\mathbf{X}_r^{(0)}$ and then followed by a convolution (convolution-1 in Fig 3) for each branch as follows:

$$\mathbf{X}_c^{(1)} = f(\mathbf{W}_c^{(1)} * \mathbf{X}_c^{(0)} + \mathbf{b}_c^{(1)}) \tag{1}$$

$$\mathbf{X}_p^{(1)} = f(\mathbf{W}_p^{(1)} * \mathbf{X}_p^{(0)} + \mathbf{b}_p^{(1)}) \tag{2}$$

$$\mathbf{X}_r^{(1)} = f(\mathbf{W}_r^{(1)} * \mathbf{X}_r^{(0)} + \mathbf{b}_r^{(1)}) \tag{3}$$

where $*$ denotes the convolution operation, $f(.)$ is an activation rectifier function [17], and the $(\mathbf{W}_c^{(1)}, \mathbf{b}_c^{(1)})$, $(\mathbf{W}_p^{(1)}, \mathbf{b}_p^{(1)})$, and $(\mathbf{W}_r^{(1)}, \mathbf{b}_r^{(1)})$ are the learnable parameters for the first layer of the three branches.

Since our objective is to have the final output size as same as the size of the input (size of the grid map), a specific type of convolution called "same convolution" is employed which allows the filter to go outside of the border of the input padding each with a zero.

Dense blocks Design. Since in our case, there is a need to capture large citywide dependencies for increasing the accuracy in predicting network demand, a deep network will be required. This will place both computational power and complexity burden on its implementation. To address this issue, DenseNet has been employed with some modifications that exploits the potential of the network through *feature reuse*, yielding condensed models that are easily trained and highly parameter efficient [13]. In our proposed ST-DenNetFus architecture, each of the outputs from the first convolution layer (shown as convolution-1 in Fig 3), $\mathbf{X}_c^{(1)}$, $\mathbf{X}_p^{(1)}$ and $\mathbf{X}_r^{(1)}$ is passed through L layers, each of which implements a non-linear transformation $\mathbf{H}_l(.)$, where l depicts the layer. In our implementation, $\mathbf{H}_l(.)$ is defined as a composite function of two consecutive operations of Rectified Linear Unit (ReLU) followed by a 3×3 convolution. On top of the L^{th} dense block, a convolutional layer is appended (shown as convolution-2 in Fig 3). The final outputs of each of these branches after convolution-2 are $\mathbf{X}_c^{(L+2)}$, $\mathbf{X}_p^{(L+2)}$ and $\mathbf{X}_r^{(L+2)}$.

4.2 External Factors & Fusion

Network demand can be affected by many complex external factors. As discussed earlier, one of the main contributions of this research is to explore if extracted urban patterns in cities can be of impact to one of the most important challenges in the telecommunications domain, Network Demand Prediction [29].

The reason behind this is that there might be a relation between the cellular data usage and external factors such as the functionality of the regions, crowd mobility and day of the week. Taking the functional regions as an example, thinking about a business district, then intuitively one could expect that most companies will be empowered by a WiFi network and hence people once they arrive to their work will probably rely on the WiFi network more than the cellular network. In contrast, in a shopping district, the cellular network might be used more than the WiFi network as usually people are walking in streets or in shops in which WiFi is not universally, freely available or has poor signal coverage. For capturing finer granularity of functional regions in our proposed model, we relied on what so-called *temporal functional regions*[5]. The concept of *temporal functional regions* has been recently introduced in [2] and shows the possibility of recognizing regions' functionalities that not only change spatially but also temporally.

Another external factor that intuitively could impact the network demand is the crowd mobility patterns as it is expected that the more crowded an area is, the higher network demand. In addition and as shown before in [26], the day of the week is of an impact to the network demand variation. The simple example is that people typically rely on their cellular networks in a different pattern on the weekends compared to the weekdays. To predict the network demand at time t, the prior three external factors: temporal functional regions, day of the week and the crowd mobility patterns can be already obtained. However, the challenge in embedding these external factors into a model is that they vary in their dimensionality. In other words, the temporal functional regions and the crowd mobility patterns are both 2-dimensional features that vary across time however, the day of the week is 1-dimensional feature that varies across the time. For addressing this challenge, various branches have been introduced in the ST-DenNetFus architecture to fuse the external features according to their dimensionality as shown in the yellow branches of Fig 3. Let $[E_{t_1}^{1D},...,E_{t_N}^{1D}]$ and $[E_{t_1}^{2D},...,E_{t_M}^{2D}]$ depict the features vectors for the 1-dimensional and 2-dimensional features respectively, where N and M indicate the number of the external 1-dimensional and 2-dimensional features respectively. For the 1-dimensional features, fully-connected layers are stacked and for the 2-dimensional features, convolutional layers with 5×5 filter are stacked for capturing the spatial dependencies of these features employing the "same convolution", for preserving the final output size to be the same as the size of the input.

After the network demand data is input and output $\mathbf{X}_{in}$ is generated, and the other branches for the external data sources $\mathbf{X}_{Ext-2D}$ and $\mathbf{X}_{Ext-1D}$ for the 2-dimensional and 1-dimensional features are produced, then a fusing layer (shown in red in Fig 3) is used. The output $\mathbf{X}_{in-Ext}$ is further fed to a *tanh* function to generate $\hat{\mathbf{X}}_t$ which denotes the predicted value at the t^{th} time interval. These operations can be summarized with the following equations:

$$\hat{\mathbf{X}}_t = tanh(\mathbf{X}_{in-Ext}) \tag{4}$$

$$\mathbf{X}_{in-Ext} = \mathbf{X}_{in} + \mathbf{X}_{Ext-2D} + \mathbf{X}_{Ext-1D} \tag{5}$$

[5] For same region, it could be classified as business district in the morning, eating in the afternoon and entertainment at night.

The ST-DenNetFus architecture can then be trained to predict $\hat{\mathbf{X}}_t$ from the Network Throughput input data and the external features by minimizing mean squared error between the predicted demand and the true demand matrix:

$$\kappa(\varepsilon) = ||\mathbf{X}_t - \hat{\mathbf{X}}_t||^2 \tag{6}$$

where ε represents all the learnable parameters in the whole ST-DenNetFu architecture.

5 Experiments setup

5.1 Dataset

The dataset used in this paper captures the application and network usage gathered from Truconnect LLC[6], a mobile service provider based in US. The raw real data contains more than 200 billion records of mobile sessions that span across 6 months starting from July 2016 until end of December 2016 in NYC. All of the mobile sessions are geo-tagged with longitude and latitude. Mobile sessions can be any type of application usage on the phone that uses mobile network. These sessions might include additional session types information such as Youtube video views, application downloads and updates, and web browsing sessions.

Each sample in the dataset is created due to one of the following: (a) every hour, (b) every change of a pixel (lat, long of 4 digits with a resolution of 10×10), (c) every application used within this pixel and hour results in a new record in the dataset. On average per mobile device, there are between $1000 - 1200$ records per day. The features that are filtered and used within this dataset are as follows:

- **Ueid:** This feature represents a mobile device unique identifier.
- **Latitude:** This value represents the latitude of the bottom-right corner of the pixel with resolution of 0.0001 degree.
- **Longitude:** This value represents the longitude of the bottom-right corner of the pixel with resolution of 0.0001 degree.
- **MaxRxThrpt:** This value represents the maximum downlink throughput observed on the network in the current pixel (in bps).
- **MaxTxThrpt:** This value represents the maximum uplink throughput observed on the network in the current pixel (in bps).

5.2 Baselines

The proposed ST-DenNetFus approach has been compared with the following 4 baselines:

[6] https://www.truconnect.com

- **Naive:** A naive model [20] works by simply setting the forecast at time t to be the value of the observation at time $t - l$ where l is the lag value. Several lag values are tested considering l equals to 1, 24, and 168 that corresponds to hourly, daily or weekly which we refer to as Naive-1, Naive-24, and Naive-168 respectively. For instance, in the case of daily, the network demand at time t on Monday is considered the same as at time t on Sunday. In the case of weekly, the network demand at time t on Monday is considered the same as at time t on previous Monday and for hourly, the network demand at time t is considered the same as at time $t - 1$. We concluded from these comparisons that Naive-1 shows the best accuracy and following that Naive-24 and Naive-168 respectively.
- **ARIMA:** An ARIMA model [33] is a well-known model for analyzing and forecasting time series data. ARIMA is considered a generalization of the simpler AutoRegressive Moving Average and adds the notion of integration. A nonseasonal ARIMA model is titled as ARIMA(p, d, q) where p is the number of autoregressive terms, d is the number of nonseasonal differences needed for stationarity, and q is the number of lagged forecast errors in the prediction equation. In our trained model, p, d, q are set to a default value with 1 [33].
- **RNN:** Recurrent Neural Networks (RNNs) [12] are a special type of neural network designed for sequence problems. Given a standard feedforward Multilayer Perceptron network, a recurrent neural network can be thought of as the addition of loops to the architecture. For example, in a given layer, each neuron may pass its signal latterly (sideways) in addition to forward to the next layer. The output of the network may feedback as an input to the network with the next input vector. And so on. In our experiments, the length of the input sequence is fixed to one of the $\{1, 3, 6, 12, 24, 48, 168\}$ and we concluded that the best accurate model is RNN-12.
- **LSTM:** The Long Short-Term Memory (LSTM) [22] network is a recurrent neural network that is trained using back propagation through time and overcomes the vanishing gradient problem. As such it can be used to create large (stacked) recurrent networks, that in turn can be used to address difficult sequence problems in machine learning and achieve state-of-the-art results [28]. Instead of neurons, LSTM networks have memory blocks that are connected into layers. The experiments are conducted on 6 LSTM variants following the same settings of RNN, including, LSTM-1, LSTM-3, LSTM-6, LSTM-12, LSTM-24, LSTM-48, LSTM-168. We concluded that LSTM-6 is the most accurate model.

Going forward in this paper, the best performing baseline models, Naive-1, RNN-12, and LSTM-6 will be referred to as simply Naive, RNN, and LSTM in the baselines comparison respectively.

6 Results

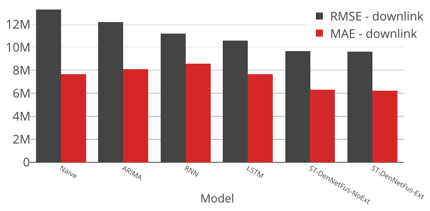
In this section, we give an overview on the preprocessing procedures taken before performing our experiments. In addition, we illustrate our procedures for selecting the tuning parameters and their effect on the prediction accuracy. Finally, we discuss how the proposed ST-DenNetFus architecture perform compared to the baselines. ST-DenNetFus is evaluated by the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) as they are two of the most common metrics used to measure the accuracy of continuous variables [6].

The learnable parameters of the ST-DenNetFus are initialized using a uniform distribution in Keras [7]. The convolutions of convolution-1 and convolution-2 use 24 and 2 filters of size 3×3 respectively. Convolution-2 uses 2 filters to match the desired number of outputs needed for the downlink and uplink throughput. Adam [16] is used for optimization, and the batch size is set to 15 for fitting the memory of the GPU used in the experiments. The number of dense blocks is set to 5. For p and r, they are empirically set to capture one-day and one-week respectively where l_c, l_p and $l_r \in \{1, 2, 3, 4\}$. From the training dataset, 90% is selected for training each model and the remaining 10% for the validation dataset which is used for choosing the best model as well as to early-stop the training algorithm if there is no improvement found after 5 consecutive epochs.

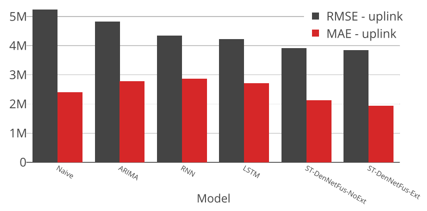
In the Dense Networks, if each function $\mathbf{H}$ produced k feature-maps as output, then the l_{th} layer will have $k \times (l-1) + k_0$ input feature-maps, where k_0 is the number of channels in the input matrix. To prevent the network from growing too wide and to improve the parameters efficiency, k is limited to a bounded integer. This hyperparamater is referred to as *growth rate* [13]. We experimented the impact of increasing the growth rate from 5 to 35 on the prediction's accuracy and we observed that the accuracy improves by increasing the growth rate until reaching certain point 24 in which widening further the network starts to have a counter impact on the accuracy. Hence it was concluded that the optimum growth rate is 24. We experimented as well the impact of the network depth and we concluded that a network depth of 5 has the optimum results to sufficiently capture with the close spatial dependence as well as the distant one. In ST-DenNetFus, each of the external features are input into a separate branch unlike the traditional approach. The traditional approach of fusing external features of same dimensionality merges these features first and then fuses them into one branch. However, in our proposed approach, a separate branch for each of the external features is used and then merge their outputs in a later stage after the feedforward execution of each of the branches (shown in yellow in Fig 3). In our case, although both the temporal functional regions and crowd mobility patterns are of same dimensionality where both are 2-dimensional matrices that vary across time (1-hour time-interval), they are each input on a separate branch and then fused later. We concluded that the impact of feeding the external features in this way performs 10% RMSE and 8% MAE better for the downlink throughput prediction and 8% RMSE and 7% MAE better for the uplink throughput prediction. In order to determine the optimum length of closeness, period and trend for the network demand dataset, the length of period and trend are set to 1 and then the length of closeness is varied from 0 to 5 where $l_c = 0$ indicates that the closeness component/branch is not employed which we concluded that l_c equals to 3 has the lowest RMSE and MAE and $l_c = 0$ has

Table 1: Prediction accuracy comparisons with baselines

	Evaluation Metric (Downlink Throughput)		**Evaluation Metric (Uplink Throughput)**	
Model	**RMSE**	**MAE**	**RMSE**	**MAE**
Naive	13278936.747	7667966.397	5237542.197	2406522.277
ARIMA	12177307.197	8073921.767	4816366.977	2783808.635
RNN	11199525.956	8576942.055	4335734.302	2865784.639
LSTM	10580656.522	7660093.113	4216037.533	2713482.051
ST-DenNetFus-NoExt	9675762.836	6315907.039	3907936.380	2131071.282
ST-DenNetFus-Ext	**9600259.526**	**6206750.047**	**3847875.555**	**1933871.466**



(a) Comparison for downlink throughput prediction accuracy.



(b) Comparison for uplink throughput prediction accuracy.

Fig. 4: Model ranking for Network Demand Prediction. The smaller the better.

the highest error. Further, l_c is set to 3 and l_r is set to 1 and then l_p is varied from 0 to 5. Then we concluded that the best performance is when l_p equals to 3. Similarly, we did for l_r and found that the best value is at 4. Based on this analysis, it is concluded that the best configuration for the $\{l_c, l_p, l_r\}$ is $\{3, 3, 4\}$.

Table 1 shows the ST-DenNetFus Network Demand Prediction accuracy comparisons with the baselines for both the throughput downlink and uplink. As shown, the proposed ST-DenNetFus consistently and significantly outperforms all baselines. Specifically, the results for the downlink throughput prediction demonstrate that ST-DenNetFus (with 5 dense-blocks) is relatively 30% RMSE and 20% MAE better than the Naive model, 20% RMSE and 23% MAE better than ARIMA, 15% RMSE and 30% MAE better than RNN and 10% RMSE and 20% MAE better than LSTM. For the uplink throughput prediction, ST-DenNetFus is 27% RMSE and 20% MAE better than the Naive model, 20% RMSE and 30% MAE better than ARIMA, 12% RMSE and 33% MAE better than RNN, and 10% RMSE and 30% MAE better than LSTM. ST-DenNetFus-NoExt is our proposed version of ST-DenNetFus-Ext that does not consider external factors (e.g. temporal functional regions). It can be seen that ST-DenNetFus-NoExt is worse than the ST-DenNetFus-Ext indicating that external factors and patterns fused are always beneficial. Intuitively, the models in RMSE can be ranked as illustrated in Fig 4.

7 Conclusion

In this paper, a new deep learning based approach called ST-DenNetFus is proposed for forecasting the network demand (throughput uplink and downlink) in each and every region of a city. For the first time, it has been shown that fusing some external patterns such as temporal functional regions and crowd mobility patterns improves the accuracy of the forecasting due to their intuitive correlation with the network demand variation. Compared to other 4 baselines, the proposed approach outperforms, confirming that the proposed approach is better and more applicable to the Network Demand Prediction problem. The introduced ST-DenNetFus is capable of learning the spatial and temporal dependencies. In addition, it employs various branches for fusing external data sources of various dimensionalities. Furthermore, we argue that the introduced ST-DenNetFus architecture could be leveraged for solving other spatio-temporal prediction problems that requires fusing external data sources such as energy demand forecasting and others. In the future, we plan to consider more external data sources such as weather data and the type of applications used by individuals in each grid cell of the city which could further boost the accuracy of the network demand prediction. In addition, we would like in the future to extend our data fusion layer from all branches using more complex techniques such as the parametric-matrix-based fusion mechanisms.

Acknowledgment

This work was partly supported by the Science Foundation Ireland ADAPT centre (Grant 13/RC/2106) and by the EC project ASGARD, 700381 (H2020-ICT-2016-09, Research and Innovation action).

References

1. Abou-Zeid, H., Hassanein, H.S.: Predictive green wireless access: Exploiting mobility and application information. IEEE Wireless Communications 20(5), 92–99 (2013)
2. Assem, H., Buda, T.S., O'sullivan, D.: Rcmc: Recognizing crowd-mobility patterns in cities based on location based social networks data. ACM Transactions on Intelligent Systems and Technology (TIST) 8(5), 70 (2017)
3. Assem, H., Ghariba, S., Makrai, G., Johnston, P., Gill, L., Pilla, F.: Urban water flow and water level prediction based on deep learning. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 317–329. Springer (2017)
4. Assem, H., O'Sullivan, D.: Discovering new socio-demographic regional patterns in cities. In: Proceedings of the 9th ACM SIGSPATIAL Workshop on Location-based Social Networks. p. 1. ACM (2016)

5. Assem, H., Xu, L., Buda, T.S., O'Sullivan, D.: Spatio-temporal clustering approach for detecting functional regions in cities. In: Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on. pp. 370–377. IEEE (2016)
6. Chai, T., Draxler, R.R.: Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. Geoscientific Model Development 7(3), 1247–1250 (2014)
7. Chollet, F.: Deep learning library for python. runs on tensorflow, theano, or cntk. `https://github.com/fchollet/keras`, [Online; accessed 09-August-2017]
8. Cisco, I.: Cisco visual networking index: Forecast and methodology, 2011–2016. CISCO White paper pp. 2011–2016 (2012)
9. Dong, X., Fan, W., Gu, J.: Predicting lte throughput using traffic time series. ZTE Communications 4, 014 (2015)
10. Hasan, Z., Boostanimehr, H., Bhargava, V.K.: Green cellular networks: A survey, some research issues and challenges. IEEE Communications surveys & tutorials 13(4), 524–540 (2011)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)
13. Huang, G., Liu, Z., Weinberger, K.Q., van der Maaten, L.: Densely connected convolutional networks. arXiv preprint arXiv:1608.06993 (2016)
14. Jain, V., Murray, J.F., Roth, F., Turaga, S., Zhigulin, V., Briggman, K.L., Helmstaedter, M.N., Denk, W., Seung, H.S.: Supervised learning of image restoration with convolutional networks. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. pp. 1–8. IEEE (2007)
15. Khan, L.U.: Performance comparison of prediction techniques for 3g cellular traffic. International Journal of Computer Science and Network Security (IJCSNS) 17(2), 202 (2017)
16. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
18. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature 521(7553), 436–444 (2015)
19. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.R.: Efficient backprop. In: Neural networks: Tricks of the trade, pp. 9–48. Springer (2012)
20. Makridakis, S., Wheelwright, S.C., Hyndman, R.J.: Forecasting methods and applications. John wiley & sons (2008)
21. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. arXiv preprint arXiv:1511.05440 (2015)
22. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech. vol. 2, p. 3 (2010)
23. Najafabadi, M.M., Villanustre, F., Khoshgoftaar, T.M., Seliya, N., Wald, R., Muharemagic, E.: Deep learning applications and challenges in big data analytics. Journal of Big Data 2(1), 1 (2015)
24. Oh, E., Krishnamachari, B., Liu, X., Niu, Z.: Toward dynamic energy-efficient operation of cellular network infrastructure. IEEE Communications Magazine 49(6) (2011)

25. Papagiannaki, K., Taft, N., Zhang, Z.L., Diot, C.: Long-term forecasting of internet backbone traffic. IEEE transactions on neural networks 16(5), 1110–1124 (2005)
26. Paul, U., Subramanian, A.P., Buddhikot, M.M., Das, S.R.: Understanding traffic dynamics in cellular data networks. In: INFOCOM, 2011 Proceedings IEEE. pp. 882–890. IEEE (2011)
27. Sadek, N., Khotanzad, A.: Multi-scale high-speed network traffic prediction using k-factor gegenbauer arma model. In: Communications, 2004 IEEE International Conference on. vol. 4, pp. 2148–2152. IEEE (2004)
28. Sahu, A.: Survey of reasoning using neural networks. arXiv preprint arXiv:1702.06186 (2017)
29. Sayeed, Z., Liao, Q., Faucher, D., Grinshpun, E., Sharma, S.: Cloud analytics for wireless metric prediction-framework and performance. In: Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on. pp. 995–998. IEEE (2015)
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
31. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
32. Waber, A.: The seasonality of mobile device usage. https://marketingland.com/seasonality-mobile-device-usage-warmer-weather-tempers-tech-95937 (2014)
33. Wu, J., Wei, S.: Time series analysis. Hunan Science and Technology Press, Chang-Sha (1989)
34. Yao, J., Kanhere, S.S., Hassan, M.: Improving qos in high-speed mobility using bandwidth maps. IEEE Transactions on Mobile Computing 11(4), 603–617 (2012)
35. Yu, Y., Song, M., Fu, Y., Song, J.: Traffic prediction in 3g mobile networks based on multifractal exploration. Tsinghua Science and Technology 18(4), 398–405 (2013)
36. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision. pp. 818–833. Springer (2014)
37. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. In: AAAI. pp. 1655–1661 (2017)
38. Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X., Li, T.: Predicting citywide crowd flows using deep spatio-temporal residual networks. arXiv preprint arXiv:1701.02543 (2017)
39. Zhou, B., He, D., Sun, Z., Ng, W.H.: Network traffic modeling and prediction with arima/garch. In: Proc. of HET-NETs Conference. pp. 1–10 (2005)